REMARKS

Claims 1, 2, 4-18, and 20-22 are pending in this application. Claims 3 and 19 are canceled. Claim 1 is currently amended. Claims 20-22 are new.

The Examiner rejected claims 1, 2 and 4-11 under 35 U.S.C. 102(e) as anticipated by U.S. Patent No. 6,950,926 issued to Menezes. Applicants respectfully traverse the Examiner's rejections.

As previously noted by Applicants, Menezes discloses a method for determining dependencies in a group of instructions executed in a processor. In particular, the disclosed method includes fetching both instructions and associated dependency instructions from a memory. The dependency instructions are obtained for instance by exploiting NOP instructions. The dependency instructions are generated by a compiler along with the assembler instructions. See Col. 1, lines 55-62 of Menezes. The dependency instructions are decoded by a decoder, then an instruction control unit and a scheduler provide for execution in the order dictated by the dependency. In an embodiment of Menezes, the dependencies are defined among single instructions in a set. See Menezes, Col. 4, lines 15-27 and Col. 4, line 63 to Col. 5, line 6. In another embodiment of Menezes, a first set and a second set can have their own internal dependencies, as discussed above, as well as a dependency between an instruction of the first set and an instruction of the second set. See Menezes, Col. 5, line 59 to Col. 6, line 15.

Claim 1 as amended recites, "[a] process for executing programs ... comprising ... compiling the program to be executed and translating said program into native instructions ... organizing the instructions deriving from the translation of said program into respective bundles arranged in an order of successive bundles, each bundle grouping together instructions adapted to be executed in parallel by said at least one processor; separating said bundles of instructions into respective sub-bundles by detecting a value of a binary symbol encoded in one of the instructions deriving from the translation of the program to be executed of the respective bundle." This means that before the separating step, the instructions are organized into bundles adapted to be executed in parallel.

The Examiner points to Col. 3, lines 24-31 of Menezes, which mentions generating an instruction with dependency information. This portion of Menezes does not

discuss organizing the instructions into bundles of instructions adopted to be executed in parallel. To the contrary, the dependency data instruction of Menezes specifies "the data dependencies among the first group of instructions." If the instructions of a group are adopted to be executed in parallel, there would be no need to "determine data dependencies among a group," because there would be no dependencies in the group, and no need for an instruction specifying "dependencies among the first group of instructions." Menezes, Col. 3, lines 26-31.

The Examiner next points to Col. 5, lines 2-4 of Menezes. This portion of Menezes indicates that when a neutral instruction indicates condition 1, then the instructions of a set can be executed concurrently. This is not the same thing as organizing the instructions of a program into bundles adopted to be executed in parallel. In the language of claim 1, it is not the same thing as "organizing the instructions ... into respective bundles ... each bundle grouping together instructions adopted to be executed in parallel." It merely reflects the fact that sometimes a group of instructions of Menezes might not have an internal dependency. The fact that Menezes is devoted (see Col. 4, lines 16-44) to provide a further instruction that explains how the instructions in a set may be executed means that the instructions are not organized into bundles of instructions adopted to be executed in parallel. Moreover, the instructions in Menezes retain the order generated by the compiler, they are not organized into bundles in a separate organizing step. Thus, Menezes does not teach, motivate or suggest organizing the instructions in bundles of instructions adopted to be executed in parallel.

In response to Applicants' prior arguments, the Examiner does not address Applicants' argument that Menezes fails to disclose organizing the instructions into bundles of instructions adopted to be executed in parallel. The Examiner instead points to Col. 5, line 63 to Col. 6, line 10. This portion of Menezes notes that two sets of instructions, in addition to having internal dependencies specified in dependency instructions in each set, may have a dependency with instructions in the other set, and that this information can be incorporated into the dependency instruction as well. The Examiner again completely overlooks the "each bundle grouping together instructions adopted to be executed in parallel" language of claim 1. The instruction sets of Menezes still have internal dependencies and require internal dependency instructions, and thus are not organized into bundles adopted to be executed in parallel.

In addition, Menezes does not teach, suggest or motivate "separating said bundles of instructions into respective sub-bundles by detecting a value of a binary symbol encoded in one of the instructions deriving from the translation of the program to be executed," as recited in claim 1. Menezes generates a new instruction containing the dependency information. Menezes, Col. 4, lines 29-62; Col. 6, lines 50-62. In Memezes, dependency information is not encoded in an existing instruction in a bundle. The embodiments of Menezes result in a 10 to 20% increase in code size, and a corresponding decrease in execution speed. In contrast, embodiments of the claimed method can encode a symbol in an existing instruction without affecting code size or execution speed. The Examiner does not address this argument in the Office Action, other than to point to Col. 5, lines 63 to Col. 6, line 10, and to Figure 1, element 12, and Col. 3, lines 24-31 of Menezes. The cited portions of Menezes do not disclose encoding dependency information in an existing instruction in a bundle of instructions.

The Examiner also appears to argue that because Menezes addresses interdependencies between instructions, it inherently organizes the instructions into bundles adopted to be executed in parallel and separates the organized bundles into sub-bundles as recited. To the extent the Examiner is making an inherency argument, Applicants respectfully traverse the Examiner's contention and request the Examiner to cite evidence that the recited organizing and separating are inherent in Menezes. Even assuming Menezes somehow discloses the recited organizing and separating (which Menezes does not), it still would not disclose the detecting a value of a binary symbol encoded in one of the instructions.

Accordingly, Applicants respectfully submit that claim 1 is not anticipated or rendered obvious by Menezes. Claims 2 and 4-11 depend from claim 1, and are allowable at least by virtue of their dependencies. Claim 3 has been canceled.

The Examiner rejected claims 12-18 under 35 U.S.C. 103(a) as rendered obvious by U.K. Patent Application No. 9725808.1 by Tulai in view of Menezes. Applicants respectfully traverse the Examiner's rejections.

Independent claim 12 recites, "organizing said instruction sets into respective groups, each group having a predetermined priority for execution in a given processor of said plurality; separating each group of instructions into a respective first sub-bundle of instructions

which must be executed before the instructions belonging to the next group, and a respective second sub-bundle of instructions that can be executed before or in parallel with respect to the instructions belonging to said next group, it being possible for at least said second sub-bundle of instructions to be a null set." The Examiner does not contend that Tulai teaches, motivates or suggests separating organized groups of instructions into sub-bundles. As noted above, the instructions in Menezes retain the order generated by the compiler, they are not organized into groups in a separate organizing step. Thus, the combination of Tulia and Menezes does not teach, suggest or motivate organizing the instructions into respective groups of instructions and further separating the respective groups into sub-bundles, as recited. Accordingly, claim 12 is not anticipated or rendered obvious by Tulia, alone or in combination with Menezes. Claims 13-15, 21 and 22 depend from claim 12 and are allowable at least by virtue of their dependencies. Claim 19 has been canceled.

Independent claim 16, as amended, recites, "a plurality of processors coupled for receiving instruction sets, each instruction set containing one or more instructions; and a first processor of the plurality coupled to an instruction stream and capable of directing said instruction sets to each of the processors of said plurality for execution; said first processor configured to direct the instructions sets to the processors of said plurality based on priority values carried by a designated number of bits encoded into each instruction." The Examiner relies on Tulia as disclosing a plurality of processors. Tulia is not an appropriate primary reference. The Examiner does not identify a processor in Tulia capable of directing the instructions sets to each of the plurality of processors. To the extent the Examiner contends the routing circuitry of Tulia (see Figure 3 of Tulia) is a processor, the routing circuitry is not capable of routing instruction sets to itself for execution. The Examiner does not address this argument in the Office Action. Accordingly, claim 16 is not anticipated or rendered obvious by Tulia, alone or in combination with Menezes. Claims 17, 18 and 20 depend from claim 16 and are allowable at least by virtue of their dependencies.

In addition, claims 15 and 18 recite, "wherein said priority is determined based on the amount of percentage of maximum power required by each of the processors of said plurality to execute said instruction set," (or similar language). The Examiner points to Tulia where the

impact of the size of an instruction register file on power consumption is discussed. There is no teaching, suggestion or motivation in Tulia to determine priority based on the maximum power required by each of the processors to execute an instruction set. Accordingly, claims 15 and 18 are not anticipated or rendered obvious over Tulia, alone or in combination with Menezes, for the additional reason that the combination of Tulia and Menezes does not teach, suggest or motivate determining priority based on the amount of percentage of maximum power required by each of the processors to execute an instruction set.

In addition, claim 20 recites, "the first processor is further configured to extract indications of sub-bundling of sets of instructions having respective priority values from the instructions and to determine, based on the extracted indications of sub-bundling, a number of instructions to be executed in each cycle." Accordingly, claim 20 is not anticipated or rendered obvious by Tulia, alone or in combination with Menezes, for the additional reason that the combination of Tulia and Menezes does not teach, suggest or motivate sub-bundling of instructions in a set of instructions having a respective priority value.

In addition, claims 21 and 22 recite, "encoding a value of a binary symbol" in an instruction, the value indicating information related to sub-bundling in a respective group of instructions. Accordingly, claims 21 and 22 are not anticipated or rendered obvious by Tulia, alone or in combination with Menezes, for the additional reason that the combination of Tulia and Menezes does not teach, suggest or motivate encoding a binary symbol indicative of sub-bundling of instructions in a group of instructions.

The Director is authorized to charge any additional fees due by way of this Amendment, or credit any overpayment, to our Deposit Account No. 19-1090.

All of the claims remaining in the application are now clearly allowable. Favorable consideration and a Notice of Allowance are earnestly solicited.

Respectfully submitted,

SEED Intellectual Property Law Group PLLC

Timothy L. Boller
Registration No. 47,435

TLB:jms

701 Fifth Avenue, Suite 5400
Seattle, Washington 98104
Phone: (206) 622-4900
Fax: (206) 682-6031

1123555_1.DOC